CERTIFIED COPY OF PRIORITY DOCUMENT

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application: ２０００年１０月２４日

出願番号
Application Number: 特願２０００－３２３８２３

出願人
Applicant(s): インターナショナル・ビジネス・マシーンズ・コーポレーション

２００１年 ６月１９日

特許庁長官
Commissioner,
Japan Patent Office

及川耕造

出証番号 出証特２００１－３０５７７４９

【書類名】　特許願
【整理番号】　JP9000274
【提出日】　平成12年10月24日
【あて先】　特許庁長官　殿
【国際特許分類】　G06F　5/00
【発明者】
　【住所又は居所】　神奈川県大和市下鶴間１６２３番地１４　日本アイ・ビー・エム株式会社　東京基礎研究所内
　【氏名】　近藤　豪
【発明者】
　【住所又は居所】　神奈川県大和市下鶴間１６２３番地１４　日本アイ・ビー・エム株式会社　東京基礎研究所内
　【氏名】　広瀬　紳一
【特許出願人】
　【識別番号】　390009531
　【氏名又は名称】　インターナショナル・ビジネス・マシーンズ・コーポレーション
【代理人】
　【識別番号】　100086243
　【弁理士】
　【氏名又は名称】　坂口　博
【代理人】
　【識別番号】　100091568
　【弁理士】
　【氏名又は名称】　市位　嘉宏
【代理人】
　【識別番号】　100106699
　【弁理士】
　【氏名又は名称】　渡部　弘道

【復代理人】

【識別番号】　100104880

【弁理士】

【氏名又は名称】　古部　次郎

【選任した復代理人】

【識別番号】　100100077

【弁理士】

【氏名又は名称】　大場　充

【手数料の表示】

【予納台帳番号】　081504

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9706050

【包括委任状番号】　9704733

【包括委任状番号】　0004480

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　構造回復システム、構文解析システム、変換システム、コンピュータ装置、構文解析方法、記憶媒体及びプログラム伝送装置

【特許請求の範囲】

【請求項１】　所定の規則に基づいて記述されたデータ列の構造を解析し、当該規則上の誤りを検出する解析手段と、

前記解析手段からの依頼に応じて、前記解析手段により検出された前記データ列における前記規則上の誤りを修正する回復手段とを備え、

前記回復手段は、

特定の種類の誤りを修正する単純な機能を有する修正手段の集合を含んで構成され、

前記データ列における前記規則上の誤りの種類に応じて前記修正手段を切り換えて使用することにより、前記データ列における種々の誤りを修正すること

を特徴とする構造回復システム。

【請求項２】　所定の規則に基づいて記述されたデータ列に対する構文解析を行う構文解析システムにおいて、

構文解析処理を実行する構文解析部と、

前記構文解析部からの依頼に応じて、前記構文解析部の構文解析処理により検出された前記データ列の誤りを修正する構文回復部とを備え、

前記構文回復部は、前記修正の内容を変更可能であること

を特徴とする構文解析システム。

【請求項３】　前記構文回復部は、前記構文解析部により検出される前記データ列の誤りの種類に応じて複数用意され、

個々の前記構文回復部は、特定の１種類の誤りを修正する一つの機能を有すること

を特徴とする請求項２に記載の構文解析システム。

【請求項４】　データ列の種類と、当該データ列の誤りを回復するために用いる前記構文回復部の組み合わせとを対応付けた対応情報を保存する対応情報保存手段をさらに備え、

前記構文解析部は、処理対象である前記データ列の種類に応じて、前記対応情報保存手段に保存された前記対応情報に基づいて、誤りの修正を依頼する前記構文回復部の組み合わせを設定すること
を特徴とする請求項３に記載の構文解析システム。

【請求項５】　前記構文回復部の少なくとも一つは、
前記データ列が前記構文解析部における構文解析処理に用いる規則において定義されていない要素を含んでいる場合に起動し、
前記構文解析部にて用いられる規則を前記要素を定義している規則に置き換え、前記データ列を前記構文解析部に戻す処理を行うこと
を特徴とする請求項３に記載の構文解析システム。

【請求項６】　処理対象である前記データ列に対する字句の解析処理を実行する字句解析部と、
前記字句解析部からの依頼に応じて、前記字句解析部の解析処理により検出された前記データ列における字句の誤りを修正する字句回復部とをさらに備え、
前記字句回復部は、前記修正の内容を変更可能であること
を特徴とする請求項２に記載の構文解析システム。

【請求項７】　前記字句回復部は、前記字句解析部により検出される前記データ列における字句の誤りの種類に応じて複数用意され、
個々の前記字句回復部は、特定の１種類の誤りを修正する一つの機能を有すること
を特徴とする請求項６に記載の構文解析システム。

【請求項８】　所定の形式で記述されたデータ列を他の形式に変換する変換システムにおいて、
前記データ列を解析する解析部と、
前記解析部からの依頼に応じて、前記解析部の解析処理により検出された前記データ列の誤りを修正する回復部と、
前記解析部の解析結果に基づいてデータ形式の変換を行う変換部とを備え、
前記回復部は、前記解析部により検出される前記データ列の誤りの種類に応じて複数用意され、

個々の前記回復部は、特定の１種類の誤りを修正する一つの機能を有することを特徴とする変換システム。

【請求項９】　前記解析部は、前記データ列に対する構文解析を行う構文解析手段であり、

前記回復部は、前記データ列における構文規則上の誤りを修正する構文回復手段であること

を特徴とする請求項８に記載の変換システム。

【請求項１０】　所定の規則に基づいて記述されたデータ列を入力する入力部と、プログラム制御により実現される機能によって当該データ列を処理する演算処理部と、当該演算処理部により処理された前記データ列を出力する出力部とを備えたコンピュータ装置において、

前記演算処理部は、

前記データ列を解析する解析部と、

前記解析部からの依頼に応じて、前記解析部の解析処理により検出された前記データ列の誤りを修正する回復部とを備え、

前記回復部は、前記解析部により検出される前記データ列の誤りの種類に応じて複数用意され、

個々の前記回復部は、特定の１種類の誤りを修正する一つの機能を有することを特徴とするコンピュータ装置。

【請求項１１】　前記解析部は、前記データ列に対する構文解析を行う構文解析手段であり、

前記回復部は、前記データ列における構文規則上の誤りを修正する構文回復手段であること

を特徴とする請求項１０に記載のコンピュータ装置。

【請求項１２】　所定の規則に基づいて記述されたデータ列に対する構文解析を行う構文解析方法において、

処理対象である前記データ列に構文規則上の誤りがあった場合に、当該誤りを修正するために使用するプログラムモジュールを選択するステップと、

前記データ列に対する構文解析を行うステップと、

構文解析において前記データ列から構文規則上の誤りが検出された場合に、前記プログラムモジュールに修正を依頼するステップと、

前記プログラムモジュールによる修正を行った後、修正された前記データ列に対する構文解析を行うステップと

を含むことを特徴とする構文解析方法。

【請求項１３】　前記使用するプログラムモジュールを選択するステップは、

処理対象である前記データ列の種類を調べるステップと、

得られた前記データ列の種類に応じて、予め定められた対応関係に基づいて使用する前記プログラムモジュールを選択するステップと

を含むことを特徴とする請求項１２に記載の構文解析方法。

【請求項１４】　前記プログラムモジュールに修正を依頼した後、当該プログラムモジュールによる指示に従って、前記データ列に対する構文解析に用いる規則を他の規則に置き換えるステップをさらに備え、

修正後のデータ列に対する構文解析を行うステップにおいて、置き換えられた前記他の規則に基づいて前記データ列に対する構文解析を行うこと

を特徴とする請求項１２に記載の構文解析方法。

【請求項１５】　コンピュータに実行させるプログラムを当該コンピュータの入力手段が読取可能に記憶した記憶媒体において、

前記コンピュータに実現させる機能として

データ列を解析する解析手段と、

前記解析手段からの依頼に応じて、前記解析手段の解析処理により検出された前記データ列の誤りを修正する回復手段とを備え、

前記回復手段は、前記解析処理により検出される前記データ列の誤りの種類に応じて複数用意され、

個々の前記回復手段は、特定の１種類の誤りを修正する一つの機能を有するプログラム・プロダクトを格納したことを特徴とする記憶媒体。

【請求項１６】　コンピュータに実現させる機能として、データ列を解析する解析手段と、前記解析手段からの依頼に応じて、前記解析手段の解析処理によ

り検出された前記データ列の誤りを修正する回復手段とを備え、前記回復手段は、前記解析処理により検出される前記データ列の誤りの種類に応じて複数用意され、個々の前記回復手段は、特定の１種類の誤りを修正する一つの機能を有するプログラム・プロダクトを記憶する記憶手段と、

前記記憶手段から前記プログラム・プロダクトを読み出して当該プログラムを送信する送信手段とを備えたことを特徴とするプログラム伝送装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、所定の規則に基づいて記述されたプログラムや文書などの字句解析及び構文解析を行い、誤りを修正する技術に関する。

【０００２】

【従来の技術】

所定の規則に基づいて記述されたプログラムや文書の誤りを修正するための技術として、当該プログラムや文書などのデータ列に対して字句解析や構文解析を行い、誤りを検出する技術がある。従来のこの種の技術では、主に、次の２種類の方法で誤りを扱っていた。

【０００３】

第１の方法は、誤りを検出した場合に警告を発し、解析を中断するか、またはその誤りの箇所よりも後方にある同期ポイントから解析を再開する方法である。すなわち、この方法では、積極的に誤りを正常な状態に回復することはしない。これは、プログラミング言語の処理系などのように解析対象の文書に誤りを許さない場合や、アプリケーションを特定しない汎用的な部品として開発された解析システムに多く採用されている手法である。

【０００４】

例として、ＸＭＬ（eXtensible Markup Language）の処理系について説明する。ＸＭＬでは、アプリケーションに依存しない汎用的な構文解析システムが、数々のベンダーから提供されている（例えば、米国オープンＸＭＬ社のOpenXMLや、米国ＩＢＭ社のXML4Jなど）。これらの処理系では、文書中に誤りがあった場

合、解析自体を諦めてしまうか、誤りを起こした字句を無視して後方の同期ポイントから再開する処理を行っている。かかる処理を行う際に、構文解析システムを操作するインターフェース（ＳＡＸ：Simple API for XML）に基づいて、ErrorHandlerという外部モジュールが、誤りの起こったポイントと誤りの説明的なメッセージを受け取ることができる。しかし、このモジュールは、単に警告を受け取っているのみであり、構文解析システムの状態や出力結果を変化させる機能は提供されていない。

【０００５】

第２の方法は、解析システムを用いているアプリケーションに対応した固有の回復方法に基づいて解析結果を出力し、解析を続行する方法である。すなわち、この方法では、単に誤りを検出するのみでなく、積極的に誤りを修正し、構文規則などの誤りのないプログラムや文書を回復する。この方法は、ソース文書の作成者と閲覧者が全く異なっており、文書中に誤りがあっても閲覧者には何らかの出力結果を生成しなければならない場合に用いられる。

【０００６】

例として、ＨＴＭＬ（HyperText Markup Language）の処理系について説明する。インターネットなどで用いられるウェブページを記述するＨＴＭＬは、通常、ＨＴＭＬ文書の作成者と閲覧者が異なる。そのため、ＨＴＭＬ文書に構文規則などの誤りがあったとしても、閲覧者に対してその誤りの存在を提示するだけでは足りず、ＨＴＭＬ文書としてのつじつまを合わせて閲覧可能な状態にする必要がある。したがって、ウェブブラウザ（ウェブページの閲覧用ソフトウェア）には、上記のような字句解析や構文解析を行う機能と、規則上の誤りが検出されたＨＴＭＬ文書を誤りのない状態に回復する機能とを有しているものがある。

【０００７】

所定のＨＴＭＬ文書中に

<P>str0 <B>str1 <I>str2</B> str3</I> str4</P>

という断片があったとする。この断片は、タグ<B> </B>と<I> </I>とが入れ子構

造になっていないため、構文規則において誤りである。しかし、ウェブブラウザに表示を行うためには、この誤りをそのまま放置することはできず、ウェブブラウザの構文解析手段は何らかの出力を生成しなければならない。

【０００８】

代表的なウェブブラウザである米国ネットスケープ・コミュニケーションズ社のNetscape Navigatorは、上記のような断片に対して

<P>str0 <B>str1 <I>str2</I> str3</B> str4</P>

のように修正する。すなわち、</B>と</I>とを入れ替えることにより入れ子構造を回復する。これにより、図１４に示すような出力結果（str1とstr3は太字、str2は太字で斜体）が得られる。

【０００９】

これに対し、他の代表的なウェブブラウザである米国マイクロソフト社のInternet Explorerは、上記のような断片に対して

<P>str0 <B>str1 <I>str2</I></B><I> str3</I> str4</P>

のように修正する。すなわち、</B>の前に</I>を補い、</B>の後に<I>を補うことにより入れ子構造を回復する。これにより、図１５に示すような出力結果（str1は太字、str2は太字で斜体、str3は斜体）が得られる。

【００１０】

図１２は、従来の構文解析システムの構成例を示す図、図１３は、図１２に示す構文解析システムにおける構文解析処理を説明するフローチャートである。

図１２を参照すると、構文解析システム１２０は、入力文書中の一定の入力ストリームを入力し字句解析を行う字句解析部１２１と、字句解析部１２１から字句を取得して構文解析を行い入力文書の構造を示す抽象構文木（ＡＳＴ）を生成して出力する構文解析部１２２と、抽象構文木の生成に用いられるノード作成部１２３とを備える。字句解析部１２１は、字句解析に用いるバッファ１２１ａと

、字句の誤りを修正する字句回復部１２１ｂとを備える。構文解析部１２２は、構文解析に用いるバッファ１２２ａ及び文脈ポインタと、構文規則の誤りを修正する構文回復部１２２ｂとを備える。また、構文解析部１２２による処理が行われる際に、構文解析に用いられる文法情報オブジェクト１２４が生成される。

【００１１】

図１３に示すように、構文解析処理が開始されると、まず、構文解析部１２２が初期化される（ステップ１３０１）。構文解析部１２２の初期化は、次の三つの作業により行われる。すなわち、①入力文書に対応する文書の種類を解析し、文法情報オブジェクト１２４を生成する。②バッファ１２２ａを空にする。③文脈ポインタが抽象構文木のルートノードを指すようにする。なお、構文解析部１２２の初期化に先立って、初期的に入力ストリームの入力及び字句解析は済んでいるものとする。

【００１２】

次に、構文解析部１２２が、バッファ１２２ａから字句を取り出し、処理対象である字句ｔとする（ステップ１３０２）。バッファ１２２ａが空の場合は（ステップ１３０１で初期化した直後は常に空の状態となっている）、字句解析部１２１に字句を要求し、取得した字句を字句ｔとする。字句ｔが入力文書の終端であった場合は、生成された抽象構文木を出力して処理を終了する（ステップ１３０３）。

【００１３】

字句ｔが入力文書の終端でない場合、構文解析部１２２は、字句ｔが文脈ポインタに対して文法的に適合するかどうかを文法情報オブジェクト１２４に問い合わせる。そして、適合するならば文脈ポインタに字句ｔを追加する（ステップ１３０４でＹｅｓ）。この追加は、次の処理によって行う。まず、非終端記号のノードｎをノード作成部１２３により作成し、文脈ポインタに追加する（ステップ１３０５）。次に、文脈ポインタの指示先を新たに追加した非終端記号のノードｎにずらす（ステップ１３０６）。そして、文脈ポインタが指し示す非終端記号のノードｎが全ての子供のノードを持った場合、文脈ポインタを親ノードへずらす（ステップ１３０７、１３０８）。文脈ポインタが指し示す非終端記号のノー

ドｎが全ての子供のノードを持っていない場合、またはステップ１３０８で文脈ポインタを親ノードへずらした後、ステップ１３０２へ戻り、次の字句を取得して同様の処理を行う。

【００１４】

ステップ１３０４において、字句ｔが文脈ポインタに対して文法的に適合しない場合、構文解析部１２２はエラーの内容を出力する（ステップ１３０９）。そして、予め定められたエラー処理を実行した後（ステップ１３１０）、ステップ１３０２へ戻り、次の字句を取得して処理を行う。ここで、エラー処理とは、当該字句ｔを読み飛ばして次の字句に対する処理に移行することや、固定的な手法による回復処理を含む。回復処理を行う場合、構文解析部１２２は、構文回復部１２２ｂを呼び出し、字句ｔが文脈ポインタに対して文法的に適合するように修正した後にステップ１３０２へ戻ることとなる。

【００１５】

このような回復処理は、構文解析システム１２０（ウェブブラウザの構文解析手段）がＨＴＭＬ専用であり、かつアプリケーション専用の独自に開発されたシステムであるために行うことができた。かかる手法を用いた構文解析システムとしては、例えば、株式会社ジャストシステムのArkや、Ｗ３Ｃ（World Wide Web Consortium）によるW3C Tidyなどがある。

なお、上記の動作例で、字句解析部１２１による字句解析の際に、必要に応じて字句回復部１２１ｂによる字句の誤りの修正が行われるが、入力ストリームの字句を所定の規則に照らして適切な字句に置き換える単純な処理であるため、説明を省略した。

【００１６】

【発明が解決しようとする課題】

上述したように、構文解析システムが特定のプログラミング言語専用であり、所定のアプリケーションに固有のシステムであれば、構文規則などに誤りのあるプログラムや文書を修正し、誤りのない状態に回復することが可能である。

【００１７】

しかし、上記のＨＴＭＬ文書の場合のように、誤りのある文書を正常な状態に

回復させる手法が複数存在する場合があるのに対し、従来の構文解析システムは、特定の誤りに対して固定的な手法で修正を行っていた。そのため、必ずしもＨＴＭＬ文書の作者の意図に応じた修正が行われるとは限らなかった。

【００１８】

さらに、文書の誤りを修正する手法は、解析システムの作成者が定義したものだけではなく、利用者自身が定義することも含め、様々な条件によって選択可能であることが望ましい。

【００１９】

そこで本発明は、プログラムや文書などのデータ列における字句や構文規則の誤りの種類に応じて複数の回復手段を用意し、選択的に適用して誤りを修正できるシステムを提供することを目的とする。

【００２０】

【課題を解決するための手段】

上記の目的を達成するため、本発明は、所定の規則に基づいて記述されたデータ列の構造を解析し、規則上の誤りを検出する解析手段と、この解析手段とは独立に設けられ、この解析手段からの依頼に応じて、この解析手段により検出されたこのデータ列における規則上の誤りを修正する回復手段とを備え、この回復手段は、特定の種類の誤りを修正する単純な機能を有する修正手段の集合を含んで構成され、このデータ列における規則上の誤りの種類に応じてこの修正手段を切り換えて使用することにより、このデータ列における種々の誤りを修正することを特徴とする。

これにより、処理対象であるデータ列の種類や誤りの種類に応じて修正手段を作成し、追加、変更しまたは不要な修正手段を削除することにより、種々のデータ列や種々の誤りに対して柔軟に対応し、適切な修正を行うことが可能となる。

【００２１】

特に、本発明は、プログラムや文書における構文解析に用いることができる。この場合、本発明は、所定の規則に基づいて記述されたデータ列に対する構文解析を行う構文解析システムにおいて、構文解析処理を実行する構文解析部と、この構文解析部からの依頼に応じて、この構文解析部の構文解析処理により検出さ

れたこのデータ列の誤りを修正する構文回復部とを備え、この構文回復部は、構文解析部とは独立に設けられ、かつ修正の内容を変更可能であることを特徴とする。

【００２２】

ここで、この構文回復部は、構文解析部により検出されるデータ列の誤りの種類に応じて複数用意される。そして、個々の構文回復部は、特定の１種類の誤りを修正する一つの機能を有する。

したがって、この構文回復部を追加、変更、削除することにより、柔軟かつ適切な誤りの修正を実現することができる。

【００２３】

ここで、適切な構文回復部を用いるための手段として、データ列の種類と、このデータ列の誤りを回復するために用いる構文回復部の組み合わせとを対応付けた対応情報を保存する対応情報保存手段をさらに備える構成とし、構文解析部は、処理対象であるデータ列の種類に応じて、この対応情報保存手段に保存された対応情報に基づいて、誤りの修正を依頼する構文回復部の組み合わせを設定することができる。

この対応情報保存手段としては、データ列の種類を識別するための識別子（ＩＤ）と、対応する構文回復部の組を登録した対応表を用いることができる。

【００２４】

さらに、この構文回復部の少なくとも一つは、処理対象のデータ列が構文解析部における構文解析処理に用いる規則において定義されていない要素を含んでいる場合に起動し、構文解析部にて用いられる規則を、このデータ列に含まれている当該要素を定義している規則に置き換え、このデータ列を構文解析部に戻す処理を行うことを特徴とする。

【００２５】

さらにまた、この構文解析システムは、字句解析においても同様の構成を取ることができる。すなわち、処理対象である前記データ列に対する字句の解析処理を実行する字句解析部と、この字句解析部からの依頼に応じて、この字句解析部の解析処理により検出されたこのデータ列における字句の誤りを修正する字句回

復部とをさらに備え、この字句回復部は、字句解析部とは独立に設けられ、かつ修正の内容を変更可能であることを特徴とする。

そして、この字句回復部は、字句解析部により検出されるデータ列における字句の誤りの種類に応じて複数用意される。また、個々の字句回復部は、特定の１種類の誤りを修正する一つの機能を有する。

これにより、字句の誤りの修正においても柔軟かつ適切な処理を行うことができる。

【００２６】

また、本発明は、上記のような構文解析システムを含むデータ変換システムとして提供することもできる。

さらに、上記のような構文解析システムやデータ変換システムを実現するコンピュータ装置として提供することもできる。

さらにまた、本発明は、上記のような構文解析システムをコンピュータにより実現させるプログラム・プロダクトとして作成し、かかるプログラム・プロダクトを記憶した記憶媒体や、ネットワークを介してこのプログラム・プロダクトを伝送する伝送装置として提供することができる。

【００２７】

また、本発明は、所定の規則に基づいて記述されたデータ列に対する構文解析を行う構文解析方法において、処理対象であるデータ列に構文規則上の誤りがあった場合に、この誤りを修正するために使用するプログラムモジュールを選択するステップと、このデータ列に対する構文解析を行うステップと、構文解析においてこのデータ列から構文規則上の誤りが検出された場合に、プログラムモジュールに修正を依頼するステップと、このプログラムモジュールによる修正を行った後、修正されたデータ列に対する構文解析を行うステップとを含むことを特徴とする。

【００２８】

ここで、この使用するプログラムモジュールを選択するステップは、処理対象であるデータ列の種類を調べるステップと、得られたデータ列の種類に応じて、予め定められた対応関係に基づいて使用するプログラムモジュールを選択するス

テップとを含む構成とすることができる。

【００２９】

さらに、この構文解析方法は、プログラムモジュールに修正を依頼した後、このプログラムモジュールによる指示に従って、構文解析に用いる規則を他の規則に置き換えるステップをさらに備え、修正後のデータ列に対する構文解析を行うステップにおいて、置き換えられた他の規則に基づいて処理対象のデータ列に対する構文解析を行う構成とすることができる。

【００３０】

【発明の実施の形態】

以下、添付図面に示す実施の形態に基づいて、この発明を詳細に説明する。

まず、本発明の概要について説明する。本発明は、プログラムや文書などの所定の規則に基づいて記述されたデータ列を他の形式に変換する場合などに用いられる構文解析システムにおいて、字句や構文規則の誤りを修正する手段を字句解析手段及び構文解析手段（以下、これらを特に区別しない場合は単に解析手段と称す）から独立させ、誤りの種類ごとの単純な修正機能を持つ回復手段群を用意する。そして、字句解析手段及び構文解析手段が誤りを発見した場合は、これら回復手段群に回復処理を依頼する。

【００３１】

処理を依頼された回復手段は、誤りの種類及び誤りを起こした字句または文字列に基づいて解析手段の出力結果及びバッファを操作する。これらの回復手段群は、解析手段に対して独立しているので、任意に追加、削除が可能である。

【００３２】

また、これらの回復手段群が行う回復処理は、お互いに独立でない場合が考えられる。すなわち、所定の文書に対して所定の回復手段の列（いくつかの回復手段の組み合わせ）により回復可能であったが、別の文書に対しては当該回復手段の列が有効ではなく、別の列を用いる必要があるといった場合である。このような場合に対処するために、文書の識別子と、それを解析するために必要な誤り回復手段の列の対応関係を保存する。そして、所定の文書に対する解析処理の実行時に、保存されている対応関係に基づいて、処理対象である当該文書に対応する

回復手段の列をシステムに設定する。

【００３３】

図１は、本発明の実施の形態における構文解析システムの全体構成を示す図である。

図１を参照すると、本実施の形態の構文解析システム１００は、入力文書中の一定の入力ストリームを入力し字句解析を行う字句解析部１０と、字句の誤りを修正する字句回復部１２と、字句解析部１０から字句を取得して構文解析を行い入力文書の構造を示す抽象構文木（ＡＳＴ）を生成して出力する構文解析部２０と、構文規則の誤りを修正する構文回復部２２と、使用する構文回復部２２を指定するために用いる構文回復部２２の対応表３０と、抽象構文木の生成に用いられるノード作成部２３とを備える。また、構文解析部２０による処理が行われる際に、構文解析に用いられる文法情報オブジェクト２４が生成される。さらに、字句解析部１０は、字句解析処理に用いるバッファ１１を備え、構文解析部２０は、構文解析処理に用いるバッファ２１及び文脈ポインタを備える。

【００３４】

図１において、字句解析部１０、字句回復部１２、構文解析部２０、構文回復部２２、ノード作成部２３及び文法情報オブジェクト２４は、コンピュータプログラムにより制御されたＣＰＵにて実現される仮想的なソフトウェアブロックである。特に、字句回復部１２及び構文回復部２２は、プログラムモジュールとして構文解析システム１００の使用者が任意に作成し、追加、変更、削除することができる。ＣＰＵを制御する当該コンピュータプログラムはＣＤ－ＲＯＭやフロッピーディスクなどの記憶媒体に格納したり、ネットワークを介して伝送したりすることにより提供される。

【００３５】

図１１は、本実施の形態の構文解析システム１００を搭載したコンピュータ装置の構成例を説明する図である。

図１１を参照すると、コンピュータ装置２００は、ＣＰＵ２１０と、ＣＰＵ２１０を制御して図１に示した構文解析システム１００の各構成要素を実現するコンピュータプログラムを格納したメモリ２２０と、処理対象である入力文書を入

力する入力部２３０と、入力文書に対する構文解析の結果を出力する出力部２４０とを備える。また、コンピュータ装置２００は、構文解析システム１００を実現するコンピュータプログラムを、上述したＣＤ－ＲＯＭやフロッピーディスクから読み出すディスクドライブ２５０や、ネットワークを介して受信する受信部２６０を備える。

このコンピュータ装置２００は、例えば、ＨＴＭＬ文書をＷＭＬ（Wireless Markup Language）文書に変換するというような、文書の形式を変換する変換システム（コンバータ）として使用され、文書形式の変換に先立って元の文書の構文解析を行う際に、構文解析システム１００を動作させる。また、コンピュータ装置２００をこのような変換システムとして用いる場合は、上述した出力部２４０の出力結果（構文解析システム１００は、本実施の形態で説明するように、入力文書の字句及び構文規則の誤りを修正するので、この出力結果は、これらの誤りが修正されたものである）に基づいて、当該入力文書の形式を変換する変換部（図示せず）をさらに備えることとなる。

【００３６】

本実施の形態の構文解析システム１００は、入力文書における構文規則の誤りを修正する手段として、構文回復部２２を、構文解析部２０から独立して設けてある。また、構文回復部２２は、特定の構文規則の誤りを修正する単純な機能を有しており、構文解析処理の際には、複数の構文回復部２２を組み合わせて用いることにより、種々の構文規則の誤りに対応する。

【００３７】

構文回復部２２の組み合わせは、処理対象である入力文書の種類に応じて決定される。入力文書の種類に応じた構文回復部２２の組み合わせの設定は、予め適当な基準を設けて行っても良いし、実際に構文解析を行った経験に基づいて行っても良い。また、入力文書の種類の分類基準は、構文解析システム１００の使用者が任意に決定することができる。

例えば、ＳＧＭＬなどのマークアップ言語を入力文書とする場合、ＳＧＭＬにおいて文書の形式を指定するＤＴＤ（Document Type Definition：文書型定義）に基づいて分類することもできる。

さらに詳細な分類を設定することも可能である。例として、ＨＴＭＬ文書（ウェブページ）を入力文書とする場合であって、特定のサイトにおいて、サイト内のＨＴＭＬ文書に対して特定の処理を行っている場合を考える。そして、ＨＴＭＬ文書に対して処理を行うプログラムの性質上、ＨＴＭＬ文書に特定の構文規則の誤りを発生させやすいものとする。この場合、当該サイト内に存在するＨＴＭＬ文書は、同様の構文規則の誤りが多いというような特定の傾向がある場合が考えられる。そこで、このサイト内のＨＴＭＬ文書を上述した入力文書の種類の分類項目としておき、これに対応する構文回復部２２の組み合わせを設定することが可能である。

設定された構文回復部２２の組み合わせは、入力文書の種類を示す文書ＩＤと対応付けて、対応表３０に登録される。そして、構文解析処理を行う際に、入力文書の種類に応じた文書ＩＤをキーとして検索することにより、適切な構文回復部２２の組み合わせを特定することができる。

【００３８】

本実施の形態では、字句回復部１２と字句解析部１０との関係も、構文回復部２２と構文解析部２０との関係と同様である。すなわち、字句回復部１２を、字句解析部１０から独立して設けてある。また、字句回復部１２は、特定の字句の誤りを修正する単純な機能を有し、字句解析処理の際には、複数の字句回復部１２を組み合わせて用いることにより、種々の字句の誤りに対応する。

ただし、字句の誤りは、ある程度、種類が限定されるため、構文回復部２２の場合のように入力文書の種類に応じた組み合わせを用意せず、入力ストリームに対して常に全ての字句回復部１２を適用するようにしても現実的な処理が可能である。したがって、本実施の形態では、対応表３０のような入力文書の種類に対する対応情報を保存する手段は設けていない。しかしながら、字句回復部１２に関しても、構文回復部２２と同様に、入力文書の種類に応じた対応情報を保存する手段を設け、入力文書の種類ごとに適切な字句回復部１２の組み合わせを特定して処理を行うこともできるのは言うまでもない。

【００３９】

次に、本実施の形態の動作について説明する。

図２は、本実施の形態の構文解析部２０による構文解析処理の流れを説明するフローチャートである。

図２に示すように、構文解析処理が開始されると、まず、構文解析部２０が初期化される（ステップ２０１）。構文解析部２０の初期化は、従来と同様に、①文法情報オブジェクト２４の生成、②バッファ２１を空にすること、③文脈ポインタが抽象構文木のルートノードを指すようにすること、の三つの作業を行い、さらに、対応表３０を参照して、入力文書の種類に基づき、対応する構文回復部２２の組を獲得し、該当する構文回復部２２を構文解析処理において使用するように設定する。

なお、構文解析部２０の初期化に先立って、初期的に入力ストリームの入力及び字句解析は済んでいるものとする。また、字句解析部１０の動作については後述する。

【００４０】

次に、構文解析部２０が、バッファ２１から字句を取り出し、処理対象である字句ｔとする（ステップ２０２）。バッファ２１が空の場合は（ステップ２０１で初期化した直後は常に空の状態となっている）、字句解析部１０に字句を要求し、取得した字句を字句ｔとする。字句ｔが入力文書の終端であった場合は、生成された抽象構文木を出力して処理を終了する（ステップ２０３）。

【００４１】

字句ｔが入力文書の終端でない場合、構文解析部２０は、字句ｔが文脈ポインタに対して文法的に適合するかどうかを文法情報オブジェクト２４に問い合わせる。そして、適合するならば文脈ポインタに字句ｔを追加する（ステップ２０４でＹｅｓ）。この追加は、次の処理によって行う。まず、非終端記号のノードｎをノード作成部２３により作成し、文脈ポインタに追加する（ステップ２０５）。次に、文脈ポインタの指示先を新たに追加した非終端記号のノードｎにずらす（ステップ２０６）。そして、文脈ポインタが指し示す非終端記号のノードｎが全ての子供のノードを持った場合、文脈ポインタを親ノードへずらす（ステップ２０７、２０８）。文脈ポインタが指し示す非終端記号のノードｎが全ての子供のノードを持っていない場合、またはステップ２０８で文脈ポインタを親ノード

へずらした後、ステップ２０２へ戻り、次の字句を取得して同様の処理を行う。

【００４２】

ステップ２０４において、字句ｔが文脈ポインタに対して文法的に適合しない場合、構文解析部２０は、エラーの内容を出力し（ステップ２０９）、構文回復部２２に当該字句ｔにおける構文規則の誤りの修正を依頼する。字句ｔにおける構文規則の誤りを修正する処理は、ステップ２０１の構文解析部２０の初期化処理において設定された構文回復部２２を順次適用して誤りの修正を試行することにより行われる。構文回復部２２を適用する順番は任意に設定可能であり、例えば適当な基準にて決定された優先度に基づいて定めることができる。

まず、未だ実行していない構文回復部２２があるかどうかを調べ（ステップ２１０）、未実行の構文回復部２２がある場合は、そのうちの一つを実行する（ステップ２１１）。そして、誤りの修正に成功した場合は、ステップ２０２へ戻り、次の字句に対する処理を行う（ステップ２１２）。

【００４３】

誤りの修正に失敗した場合、ステップ２１０に戻り、未実行の構文回復部２２を選んで実行する。ステップ２１０において、未実行の構文回復部２２が存在しない場合、当該入力文書に対応するとして用意された構文回復部２２によっては当該字句ｔにおける構文規則の誤りを修正することができなかったので、予め定められたエラー処理を実行した後（ステップ２１３）、ステップ２０２へ戻る。ここで、エラー処理とは、当該字句ｔを読み飛ばして次の字句に対する処理に移行することや、構文解析部２０自身が文脈ポインタに適当な字句ｔを付与するといった処理である。

【００４４】

上記の動作において、各構文回復部２２は、次のようにして誤りの修正を実行する。

すなわち、まず、構文解析部２０から、ステップ２０４の解析により得られた誤りの種類及び処理対象である字句ｔを付加情報として受け取る。この付加情報に基づいて、修正可能かどうかを判断する。修正不可能と判断したならば、次の構文回復部２２に処理を渡し、次の構文回復部２２が存在しないときは修正不可

能であることを構文解析部２０に通知する。

一方、修正可能である場合は、それまでに生成されている抽象構文木、構文解析部２０のバッファ２１及び文脈ポインタを設定し直すことにより、当該字句ｔに関する箇所を、構文規則の誤りがない状態に回復させる。そして、修正が成功したことを構文解析部２０に通知する。

【００４５】

図３は、本実施の形態の字句解析部１０による字句解析処理の流れを説明するフローチャートである。

図３に示すように、字句解析処理が開始されると、まず、字句解析部１０が初期化される（ステップ３０１）。字句解析部１０の初期化は、バッファ１１を初期化する作業により行われる。また、入力文書の種類により使用する字句回復部１２を選択する場合は、この初期化の時点で、対応表３０などを参照して使用する字句回復部１２を設定する。

【００４６】

次に、字句解析部１０は、構文解析部２０から字句を要求されるのを待って、バッファ１１から文字列strを取り出して字句化し、構文解析部２０に渡す（ステップ３０２、３０３、３０４）。バッファ１１が空の場合は（ステップ３０１で初期化した直後は常に空の状態となっている）、字句解析部１０が入力文書の入力ストリームから文字列strを読込み、解析して構文解析部２０に渡す。字句を構文解析部２０に渡した後、字句解析部１０はステップ３０２に戻り、構文解析部２０から字句を要求されるのを待って、次の文字列strに対する処理を行う。

【００４７】

ステップ３０３において、文字列strを字句化できなかった場合、字句回復部１２を順次適用して修正処理を行う（ステップ３０５、３０６、３０７）。字句回復部１２は、文字列strを正しい文字列に置き換え、バッファ１１に挿入し、修正が成功したことを字句解析部１０に通知する。そして、字句解析部１０はステップ３０２に戻り、構文解析部２０から字句を要求されるのを待って、次の文字列strに対する処理を行う。

【００４８】

全ての字句回復部１２によっても文字列strの修正が成功しなかった場合、最後の字句回復部１２が、修正が成功しなかったことを字句解析部１０に通知し、字句解析部１０は、予め定められたエラー処理を実行した後（ステップ３０８）、ステップ３０２へ戻る。ここで、エラー処理とは、当該文字列strを無視して次の文字列に対する処理に移行することなどである。

【００４９】

次に、本実施の形態による具体的な誤りの修正例について説明する。

ここでは、本実施の形態による具体的なシステムとして、ウェブページ上のコンテンツをオブジェクトとして扱うためのＤＯＭ（Document Object Model）を生成するＳＧＭＬパーザー（Parzer：構文解析装置）に基づいたＨＴＭＬパーザーを取り上げる。ＳＧＭＬのようなマークアップ言語では、構文解析システム１００によって検出されるエラー（構文規則上の誤り）には、以下のようなものがある。ここでは、これを構文解析部２０が構文回復部２２に渡す誤りの種類とする。

【００５０】

１．未定義属性

例えば、

<BODY background="bg.gif">

という開始タグがあった場合、background属性はＨＴＭＬ４．０ Strictでは定義されていないため、エラーとなる（ＨＴＭＬ３．２やＨＴＭＬ４．０ Transitional には定義されている）。

【００５１】

２．非対応終了タグ

（１）本来あるべき位置よりも前方にある場合

例えば、

str0 str1 str2 str3 str4

という断片で、入れ子構造をなすためには、はよりも後方にあるべきであるため、エラーとなる。

（２）対応すべき開始タグがない場合

例えば、

<table>

<td>..</td> <td> .. </td> </tr>

</table>

という断片において、</tr>に対する開始タグ<tr>がないため、エラーとなる。

また、

str0 str1 str2 str3 str4

という断片において、に対応する開始タグがあった場合でも、を取り扱う構文回復部２２がの直前にがあると解釈した場合、str3の直後のに対応する開始タグが無いこととなるため、エラーとなる。

【００５２】

３．不正エレメント（文脈ポインタの子供として文法的に正しくないエレメント）

例えば、

<table>

<form>

<tr><td>..</td></tr>

<tr><td>..</td></tr>

</form>

</table>

という断片において、TABLEの直下にはFORMは入らないという規則があるため、エラーとなる。

また、

. . . . .

 AnotherPage

. . . . .

という断片において、アンカーエレメント（タグ<a>）の下にあるアンカーエレメントは例外として認められないという規則があるため、エラーとなる。

【００５３】

４．未定義エレメント

例えば、FONTエレメントはＨＴＭＬ４．０ Strictでは定義されていないため、エラーとなる（ＨＴＭＬ３．２やＨＴＭＬ４．０ Transitional には定義されている）。

【００５４】

次に、構文規則の誤りを含む入力文書の例を挙げて、誤りを修正する動作を説明する。

ここでは、誤りとして、上述した２．非対応終了タグの（１）本来あるべき位置よりも前方にある場合の誤りと、（２）対応すべき開始タグがない場合の誤りとをそれぞれ含む、下記の文書Ａと文書Ｂを例として取り上げる。なお、文書Ａ、Ｂにおいて、誤り２．（１）発生、誤り２．（２）発生とコメントのある箇所が、それぞれ上述した誤りを含む箇所である。

【００５５】

文書Ａ：

```
<table>
 <tr>
  <td>
   <form action="./invention1.html">
    <input name="input1" size=5>
  </td>      <!-- 誤り２．（１）発生 -->
  <td>
    <input type=submit>
   </form>      <!-- 誤り２．（２）発生 -->
  </td>
 </tr>
```

```
</table>
```

【００５６】

文書Ｂ：

```
<table>
<tr><td><form action="./invention1.html"><input name="input1" size
=5><input type=submit></td></tr>
<tr>
 <form>
  <td>
   <select>
    <option>
     . . . . .
    <option>
   </select>
  <td>
 </form>
</tr>
</table>
<table>
<tr>
 <td>
  <select>
   <option>
    . . . . .
   <option>
  </select>
 </td>
</tr>
</table>
```

</form>　　<!-- 誤り２．（２）発生 -->

【００５７】

図４は、上記文書Ａに対して構文回復部２２による誤りの修正を行わずに構文解析部２０による構文解析を行った場合における出力結果の抽象構文木を示す図である。

図４を参照すると、２番目のINPUTエレメントがFORMエレメントの下に位置していないという誤りが発生している。これは、文書Ａにおける１番目の</td>タグが出現した場合にそれに対応するTDエレメントを閉じた結果、FORMエレメントも同時に閉じてしまうという構文回復部２２による修正なしの場合のデフォルト処理が原因となっている。

【００５８】

図５は、上記文書Ｂに対して構文回復部２２による誤りの修正を行わずに構文解析部２０による構文解析を行った場合における出力結果の抽象構文木を示す図である。

図５を参照すると、これ２番目のSELECTエレメントがFORMの下に位置していないという誤りが発生している。これは、文書Ｂにおける最後の</form>タグに対応する開始タグ<form>が存在しないため、</form>タグを無視して処理を終了したことが原因となっている。

【００５９】

これらのFORMタグに関する誤りに対し、構文回復部２２として、FormExpanderとFormInserterとを用いて誤りを修正する場合を考える。FormExpander及びFormInserterの起動条件と誤りを修正する動作（回復動作）は以下に示す通りである。

【００６０】

〔FormExpander〕

起動条件：誤りを起こした字句ｔがFORMエレメントの終了タグ</form>であり、現在の文脈ポインタが指し示すノードからルート方向に辿ってもFORMエレメントが見つからず、かつ当該字句ｔが文脈ポインタの前方にある場合。

回復動作：

１．文脈ポインタから前方へ走査し、FORMエレメントfe0を探す。

２．FORMエレメントfe0を元々あった位置p0から除く。ここで言う「除く」とは、当該ノードが保持している子孫のノードは削除せずにFORMエレメントfe0のみを取り除ことであり、当該子孫のノードはFORMエレメントfe0の親ノードに追加する。

３．文脈ポインタと位置p0とを最小の範囲で覆うことができるような位置にFORMエレメントfe0を挿入する。

４．文脈ポインタがFORMエレメントfe0の親ノードを指すようにする。

【００６１】

〔FormInserter〕

起動条件：誤りを起こした字句ｔがFORMエレメントの終了タグ</form>であり、現在の文脈ポインタが指し示すノードからルート方向に辿ってもFORMエレメントが見つからず、かつ当該字句ｔが文脈ポインタの前方にある場合。

回復動作：

１．文脈ポインタの前方へFORMエレメントfe1を探す。

２．FORMエレメントfe2を作る。

３．文脈ポインタの指し示すノードと共通の親ノードを持つ他のノードのなかで、FORMエレメントfe1よりも後方にあるものを、文脈ポインタの指し示すノードの親ノードcp0から切り取り、FORMエレメントfe2に子ノードとして追加する。ただし、ここで言う「切り取る」とは、上述した「除く」と違い、当該ノードが保持している子孫のノードも含めた部分木を取り除くことである。

４．FORMエレメントfe2をノードcp0に追加する。

５．文脈ポインタがノードcp0を指すようにする。

【００６２】

上述したように、FormExpanderとFormInserterとは、起動条件が同一である。すなわち、構文解析において、入力文書から上記の起動条件に合致する構文規則の誤りが検出された場合、FormExpanderまたはFormInserterのいずれか一方を選択的に用いることにより、当該誤りを修正することが可能となる。

【００６３】

図６は、文書Ａに対してFormExpanderを用いて誤りを修正した場合における構文解析部２０による構文解析の出力結果の抽象構文木を示す図である。

図６を参照すると、TRエレメントの下にFORMエレメントが来るという問題はあるが、二つのINPUTエレメントがそれぞれFORMエレメントの下に位置していることがわかる。

【００６４】

図７は、文書Ｂに対してFormExpanderを用いて誤りを修正した場合における構文解析部２０による構文解析の出力結果の抽象構文木を示す図である。

図７を参照すると、２番目のSELECTエレメントはFORMの下に位置させることができた。しかし、FORMの下にFORMが位置する例外に該当してしまっている。

【００６５】

図８は、文書Ａに対してFormInserterを用いて誤りを修正した場合における構文解析部２０による構文解析の出力結果の抽象構文木を示す図である。

図８を参照すると、本来は同じFORMエレメントの下に位置しなければならない二つのINPUTエレメントが、別々のFORMエレメントの下に位置している。

【００６６】

図９は、文書Ｂに対してFormInserterを用いて誤りを修正した場合における構文解析部２０による構文解析の出力結果の抽象構文木を示す図である。

図９を参照すると、２番目のSELECTエレメントがFORMの下に位置しており，かつFORMエレメントがFORMエレメントを含んでいないことがわかる。

【００６７】

以上のFormExpander及びFormInserterの適用結果を参酌すると、文書Ａに示す誤りを修正するにはFormExpanderが適しており、文書Ｂに示す誤りを修正するにはFormInserterが誤り回復器として適していることがわかる。したがって、文書Ａを構文解析する場合と、文書Ｂを構文解析する場合とで、使用する構文回復部２２を変更することが好ましい。

【００６８】

図１０は、上記の動作例において、適切な構文回復部２２を適用するための対応表３０の構成例を示す図表である。

図１０を参照すると、文書の識別子として「文書Ａ」、「文書Ｂ」が登録され（実際には、文書Ａ、Ｂを識別する所定のＩＤが登録される）、「文書Ａ」に対応する構文回復部２２としてFormExpanderが登録され、「文書Ｂ」に対応する構文回復部２２としてFormInserterが登録されている。

これにより、文書Ａを構文解析する場合には、構文解析部２０を初期化した際に、文書ＡのＩＤをキーとして構文回復部２２としてFormExpanderを使用するように設定される。同様に、文書Ｂを構文解析する場合には、構文解析部２０を初期化した際に、文書ＢのＩＤをキーとして構文回復部２２としてFormInserterを使用するように設定される。そして、適切な構文回復部２２を用いて入力文書（文書Ａまたは文書Ｂ）の誤りを修正することができ、最適な出力結果（抽象構文木）を得ることができる。

【００６９】

以上、FORMタグに関する誤りを修正する構文回復部２２について説明したが、次に、他のいくつかの構文回復部２２について例を挙げて説明する。なお、以下に示す構文回復部２２は、ＨＴＭＬ文書に対する構文解析に用いられる構文回復部２２の例であるが、ＨＴＭＬ文書以外のＳＧＭＬ文書やＸＭＬ文書にそのまま適用したり、一般化して適用したりできるものもある。

【００７０】

〔DefaultErrorHandler〕

起動条件：誤りの種類が不正エレメント（上述したエラーの種類の３）であり、誤りを起こしたエレメントの正しい出現場所が固定的である場合。ここで、固定的とは、ＤＴＤ（Document Type Definition：文書型定義）によって、エレメントの位置が繰り返しのない親エレメントの下に固定されていることを意味する。例えば、HTMLエレメントの下のHEADエレメントなどが該当する。

回復動作：

１．不正エレメントe1と同名のエレメントe2を探す。

２．エレメントe2が他の構文回復部２２の処理または構文解析部２０の処理によって補完されたエレメントかどうかを調べる。

３．エレメントe2が補完されたエレメントである場合、エレメントe2をエレメン

トe1で置き換える。

この構文回復部２２は、ＨＴＭＬとは独立であるので、ＳＧＭＬ文書やＸＭＬ文書に適用することができる。

【００７１】

〔InterleavedEndtagExchanger〕

起動条件：誤りの種類が非対応終了タグであり、当該タグが本来あるべき位置よりも前方にある場合（上述したエラーの種類の２．（１））。

回復動作：

１．エレメントを示す変数e1に文脈ポインタが指し示すノードの親ノードを代入する。

２．変数e1が、誤りを起こした非対応終了タグet0と同じである場合、修正は不可能と判断し、処理を終了する（回復失敗）。

３．変数e1が非対応終了タグet0と同じでなく、終了タグを省略不可能である場合、構文解析部２０のバッファ２１から後方に向かって、変数e1と同名の終了タグet1を探索する。

４．ステップ３の探索により終了タグet1を発見できた場合、構文解析部２０のバッファ２１におけるエレメントet1が入っていた位置に、非対応終了タグet0を入れ、バッファ２１の先頭に終了タグet1を入れる。

５．ステップ３の探索で終了タグet1を発見できなかった場合、変数e1を当該変数e1の親ノードに変更してステップ２へ戻る。

この構文回復部２２は、ＨＴＭＬとは独立であるので、ＳＧＭＬ文書やＸＭＬ文書に適用することができる。

【００７２】

〔RangeExpander〕

起動条件：誤りの種類が非対応終了タグであり、当該タグが本来あるべき位置よりも前方にある場合（上述したエラーの種類の２．（１））。

回復動作：

１．エレメントを示す変数e1に文脈ポインタを代入し、エレメントの配列型の変数配列を作成する。

２．変数e1が誤りを起こした非対応終了タグet0と同じ名前かどうかを判断し、同じでなければ同じ名前になるまで、以下のステップ３、４を繰り返す。

３．変数e1が終了タグを省略不可能である場合、ノード作成部２３を用いて変数e1と同名のエレメントを作成し、ステップ１で作成した変数配列の末尾に挿入する。

４．変数e1をエレメントe1にずらしてステップ２へ戻る。

５．ステップ２において、変数e1が誤りを起こした非対応終了タグet0と同じ名前であるならば、変数配列が空かどうかを判断し、空でなければ空になるまでステップ６を繰り返す。

６．変数配列の先頭からエレメントe2を取り出し、文脈ポインタにエレメントe2を追加する。そして、文脈ポインタの指示先をエレメントe2に変更する。

この構文回復部２２は、ＨＴＭＬとは独立であるので、ＳＧＭＬ文書やＸＭＬ文書に適用することができる。

【００７３】

〔AnchorUnderAnchorHandler〕

起動条件：誤りの種類が不正エレメント（上述したエラーの種類の３）であり、誤りを起こしたエレメントa1がアンカーであり、文脈ポインタが指し示すノードの上方にもアンカーであるエレメントa2がある場合。

回復動作：エレメントa1を、エレメントa2と共通の親ノードを持つノードとして追加し、文脈ポインタの指示先をエレメントa1に変更する。

この構文回復部２２は、上方エレメントueの下で下方エレメントdeが例外となる誤りの場合に、上方エレメントdeを下方エレメントueと共通の親を持つノードにするという回復方法として一般化できる。これにより、ＳＧＭＬ文書やＸＭＬ文書にも適用することができる。

【００７４】

〔FramesetErrorHandler〕

起動条件：誤りの種類が未定義エレメント（上述したエラーの種類の４）であり、誤りを起こしたエレメントe1がFRAMESETである場合。

回復動作：

１．エレメントe1を構文解析部２０のバッファ２１に戻す。

２．構文解析部２０が使用している文法情報オブジェクト２４を、FRAMESETを定義しているＨＴＭＬ４．０ Framesetなどに変更する。

この構文回復部２２は、未定義エレメントe2に対し、エレメントe2を定義している文書型定義DTD2に文法情報オブジェクト２４を設定する方法として一般化できる。これにより、ＳＧＭＬ文書やＸＭＬ文書にも適用することができる。文書型定義DTD2は、使用者により明示的に指定しても良いし、ＤＴＤのレポジトリなどから探索しても良い。

【００７５】

〔HTMLErrorHandler〕

起動条件：誤りの種類が不正エレメント（上述したエラーの種類の３）であり、誤りを起こしたエレメントe1の名前が

LINK, STYLE, META, BASE, ISINDEX

のいずれかである場合。

回復動作：出力結果ASTのルートノードにあるHTMLエレメントの子ノードからHEADエレメントを探し、当該HEADエレメントのノードにエレメントe1を追加する。

この構文回復部２２は、ＨＴＭＬに固有のものである。

【００７６】

〔IgnoreFont〕

起動条件：誤りの種類が不正エレメント（上述したエラーの種類の３）であり、誤りを起こしたエレメントe1の名前がFONTである場合。

回復動作：何ら処理を行わず、エレメントe1を無視する。

この構文回復部２２は、一般化してＳＧＭＬ文書やＸＭＬ文書にも適用することができる。

【００７７】

〔TRErrorHandler〕

起動条件：誤りの種類が不正エレメント（上述したエラーの種類の３）であり、誤りを起こしたエレメントe1の名前がTDである場合。

回復動作：文脈ポインタの指示先がTBODYエレメントまたはTABLEエレメントであ

る場合に、新たにTRエレメントtrを作成し文脈ポインタに追加する。そして、エレメントtrに対してエレメントelを追加し、文脈ポインタの指示先をエレメントelに設定する。

【００７８】

次に、字句回復部１２の例を説明する。字句解析においては、文字列が予め定められたタグの字句などに該当するかどうかを調べるだけなので、字句回復部１２の種類は多くはない。以下に、代表的な例を挙げる。

〔AttributeValueErrorHandler〕

起動条件：開始タグ中の属性値を字句解析中に、

"... > ... < ..."

という文字列が出現した場合。

回復動作："... "> ... < ..."という文字列を字句解析部１０のバッファ１１に挿入する。

【００７９】

なお、上記の説明では、処理対象となる入力文書がマークアップ言語で記述された文書、特にＨＴＭＬ文書である場合について説明したが、他のプログラミング言語で記述されたプログラムや自然言語で記述された文書に対する構文解析にも、当該プログラミング言語や自然言語における字句の規則及び構文規則に対応する回復手段を用意することにより、そのまま利用することが可能である。

【００８０】

さらに、これらのプログラムや文書に限らず、楽譜のような一定の規則に基づいて記述されたデータ列に対する構造解析及び誤りの修正を行う構造回復システムとして利用することもできる。

この場合、当該構造回復システムは、データ列の構造を解析し、規則上の誤りを検出する解析手段（本実施の形態における構文解析部２０に相当する）と、この解析手段とは独立に設けられ、解析手段からの依頼に応じて、データ列から検出された規則上の誤りを修正する回復手段（本実施の形態における構文回復部２２の集合に相当する）とを備え、かつこの回復手段は、特定の種類の誤りを修正する単純な機能を有する修正手段（本実施の形態における個々の構文回復部２２

に相当する）の集合を含んで構成される。そして、データ列における規則上の誤りの種類に応じてこの修正手段を切り換えて使用することにより、このデータ列における種々の誤りを修正することとなる。

【００８１】

【発明の効果】

以上説明したように、本発明によれば、プログラムや文書などのデータ列における字句や構文規則の誤りの種類に応じて複数の回復手段を用意し、選択的に適用して誤りを修正することができる。

【図面の簡単な説明】

【図１】　本発明の実施の形態における構文解析システムの全体構成を示す図である。

【図２】　本実施の形態の構文解析部による構文解析処理の流れを説明するフローチャートである。

【図３】　本実施の形態の字句解析部による字句解析処理の流れを説明するフローチャートである。

【図４】　構文回復部による誤りの修正を行わずに構文解析部による構文解析を行った場合における出力結果の抽象構文木の例を示す図である。

【図５】　構文回復部による誤りの修正を行わずに構文解析部による構文解析を行った場合における出力結果の抽象構文木の他の例を示す図である。

【図６】　構文回復部を用いて誤りを修正した場合における構文解析部による構文解析の出力結果の抽象構文木の例を示す図である。

【図７】　構文回復部を用いて誤りを修正した場合における構文解析部による構文解析の出力結果の抽象構文木の他の例を示す図である。

【図８】　他の構文回復部を用いて誤りを修正した場合における構文解析部による構文解析の出力結果の抽象構文木の例を示す図である。

【図９】　他の構文回復部を用いて誤りを修正した場合における構文解析部による構文解析の出力結果の抽象構文木の他の例を示す図である。

【図１０】　本実施の形態において適切な構文回復部を適用するために用いる対応表の構成例を示す図表である。

【図１１】　本実施の形態の構文解析システムを搭載したコンピュータ装置の構成例を説明する図である。

【図１２】　従来の構文解析システムの構成例を示す図である。

【図１３】　図１２に示す従来の構文解析システムにおける構文解析処理を説明するフローチャートである。

【図１４】　従来の構文解析システムによるＨＴＭＬ文書の修正例を示す図である。

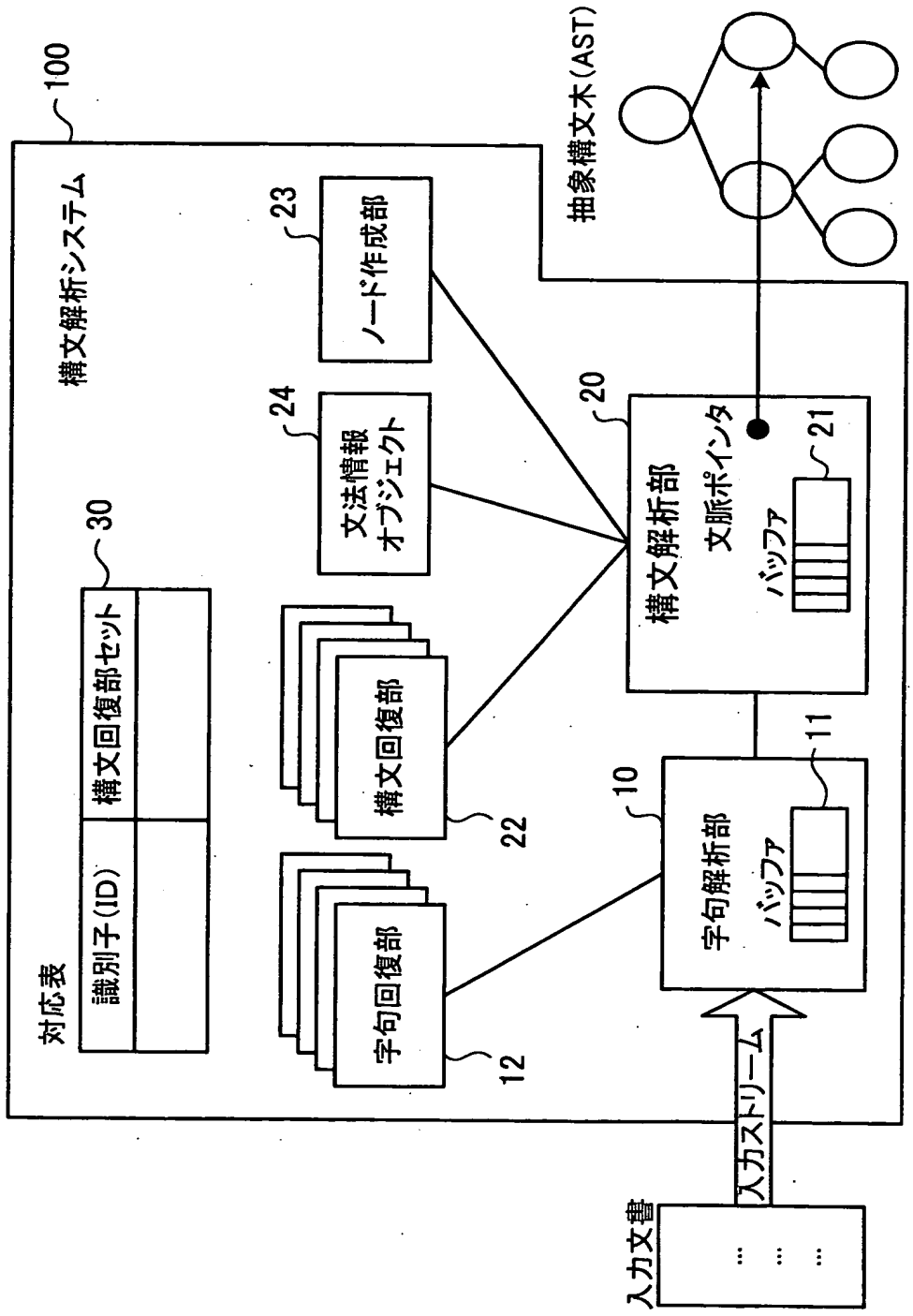
【図１５】　従来の構文解析システムによるＨＴＭＬ文書の他の修正例を示す図である。

【符号の説明】

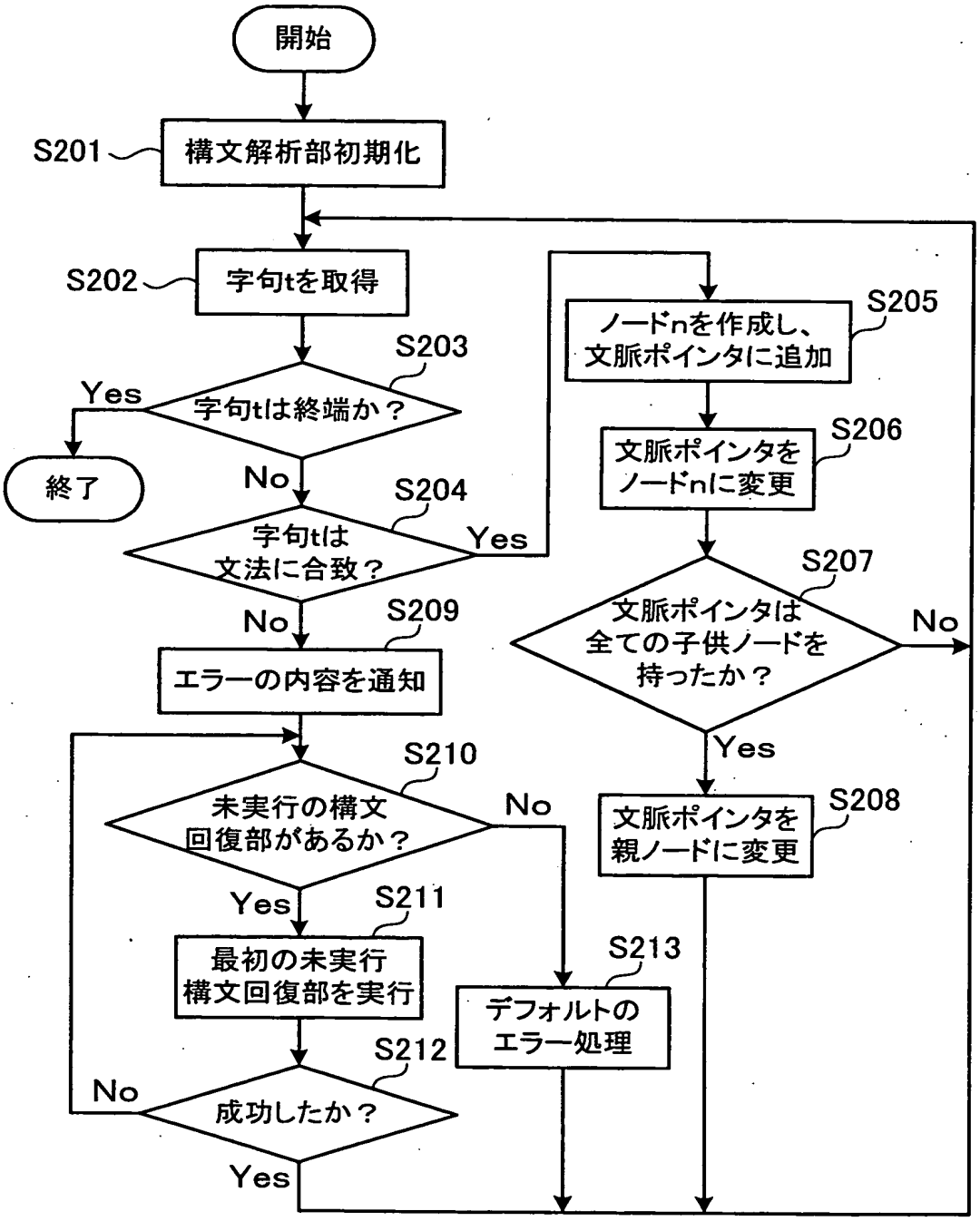
１０…字句解析部、１１、２１…バッファ、１２…字句回復部、２０…構文解析部、２２…構文回復部、２３…ノード作成部、２４…文法情報オブジェクト、３０…対応表

【書類名】　　図面

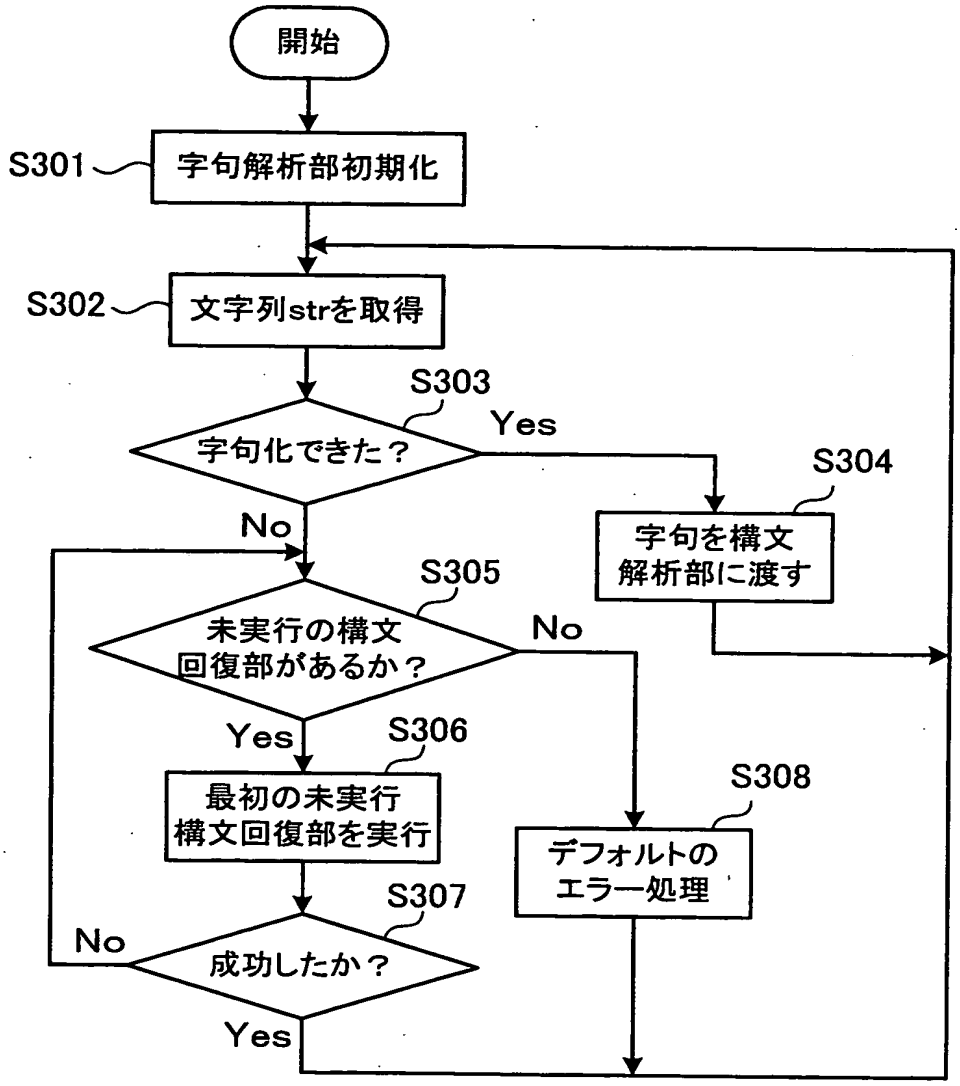
【図１】


100
構文解析システム
抽象構文木(AST)
23
ノード作成部
24
文法情報
オブジェクト
20
構文解析部
文脈ポインタ
21
バッファ
30
構文回復部セット
識別子(ID)
対応表
構文回復部
22
字句回復部
12
10
字句解析部
バッファ
11
入力ストリーム
入力文書

【図２】

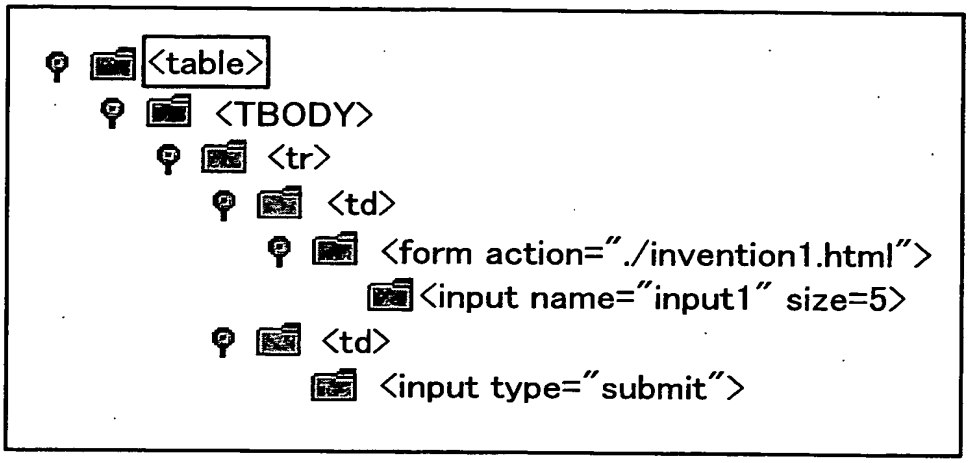

開始
S201
構文解析部初期化
S202
字句tを取得
S203
字句tは終端か？
Yes
終了
No
S204
字句tは
文法に合致？
Yes
No
S209
エラーの内容を通知
S210
未実行の構文
回復部があるか？
No
Yes
S211
最初の未実行
構文回復部を実行
S212
成功したか？
No
Yes
S213
デフォルトの
エラー処理
S205
ノードnを作成し、
文脈ポインタに追加
S206
文脈ポインタを
ノードnに変更
S207
文脈ポインタは
全ての子供ノードを
持ったか？
No
Yes
S208
文脈ポインタを
親ノードに変更

【図３】


開始
S301
字句解析部初期化
S302
文字列strを取得
S303
字句化できた？
Yes
S304
字句を構文
解析部に渡す
No
S305
未実行の構文
回復部があるか？
No
Yes
S306
最初の未実行
構文回復部を実行
S308
デフォルトの
エラー処理
S307
No
成功したか？
Yes

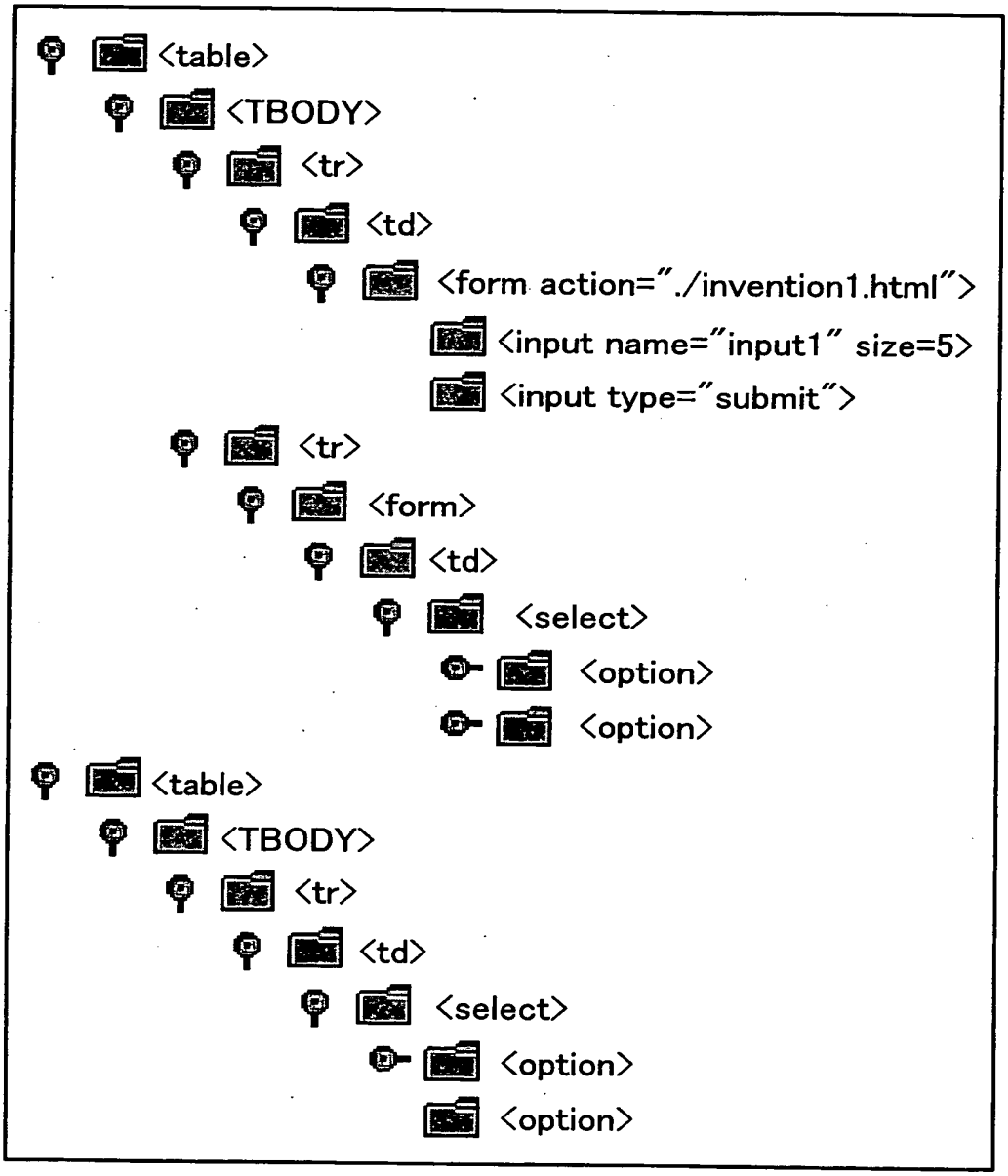
【図４】

構文回復部による修正を行わない場合の文書Aの解析結果



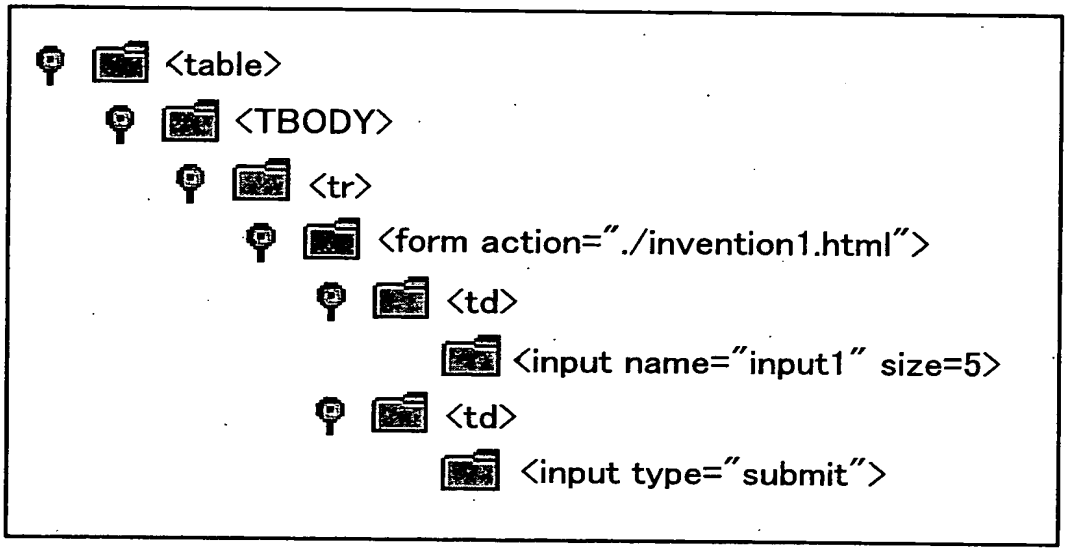
【図５】

構文回復部による修正を行わない場合の文書Bの解析結果



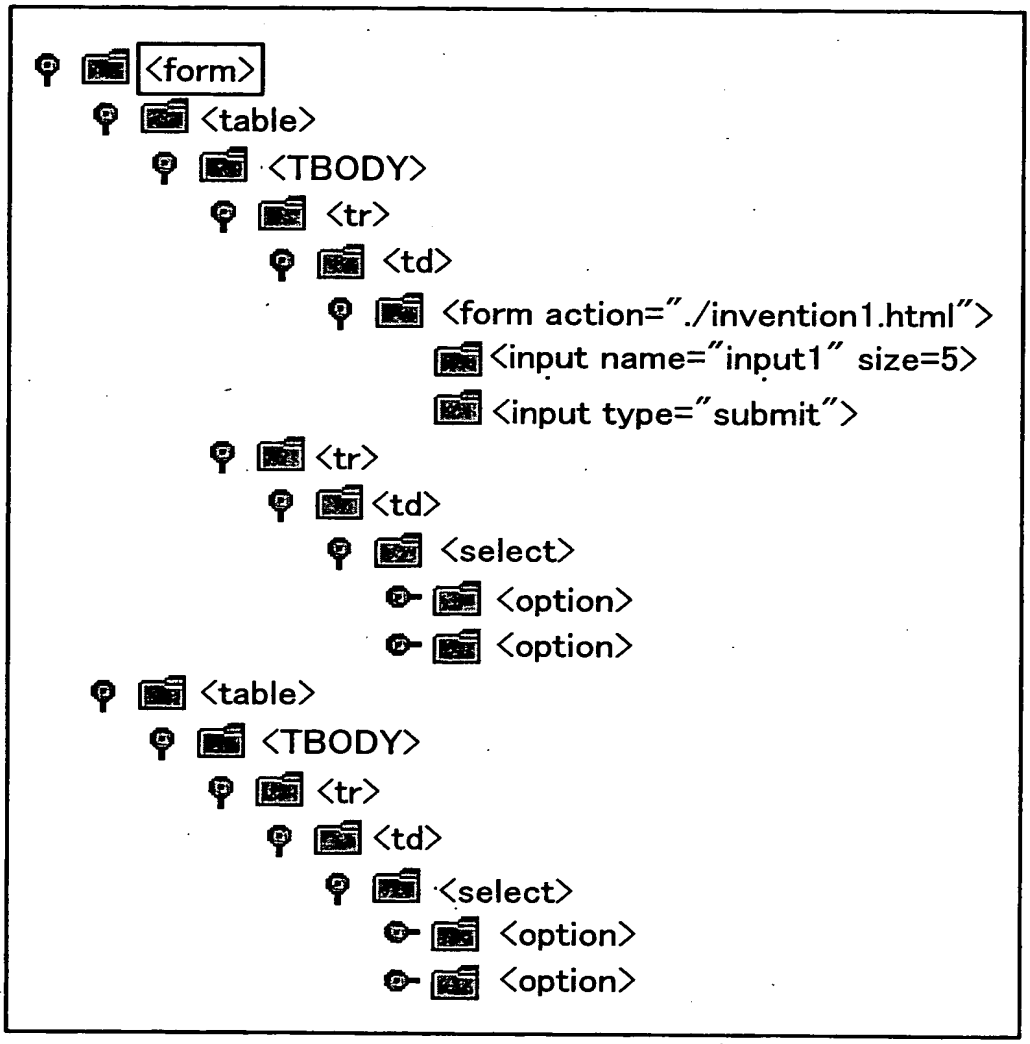
【図６】

FormExpanderを適用した場合の文書Aの解析結果



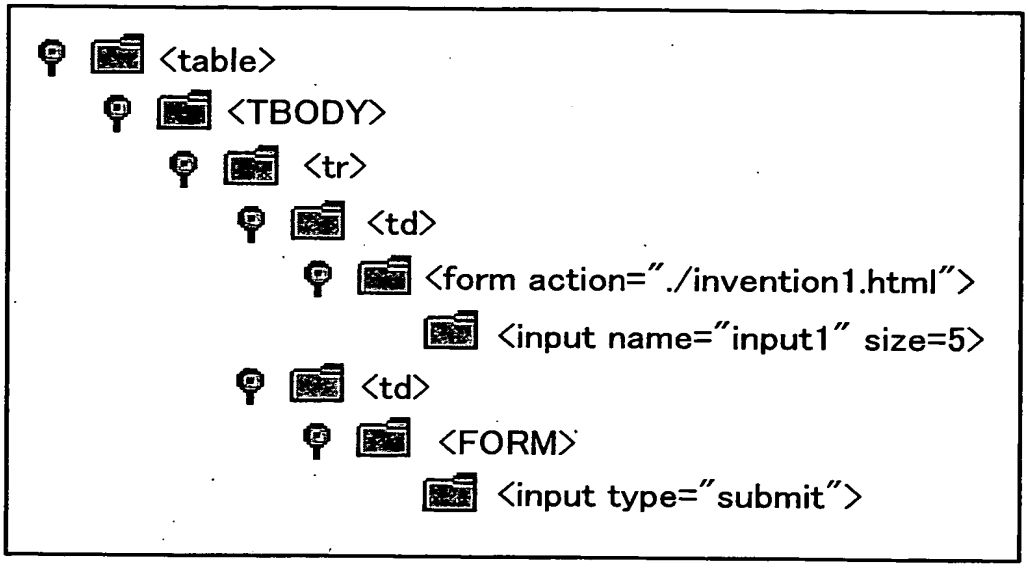
【図７】

FormExpanderを適用した場合の文書Bの解析結果



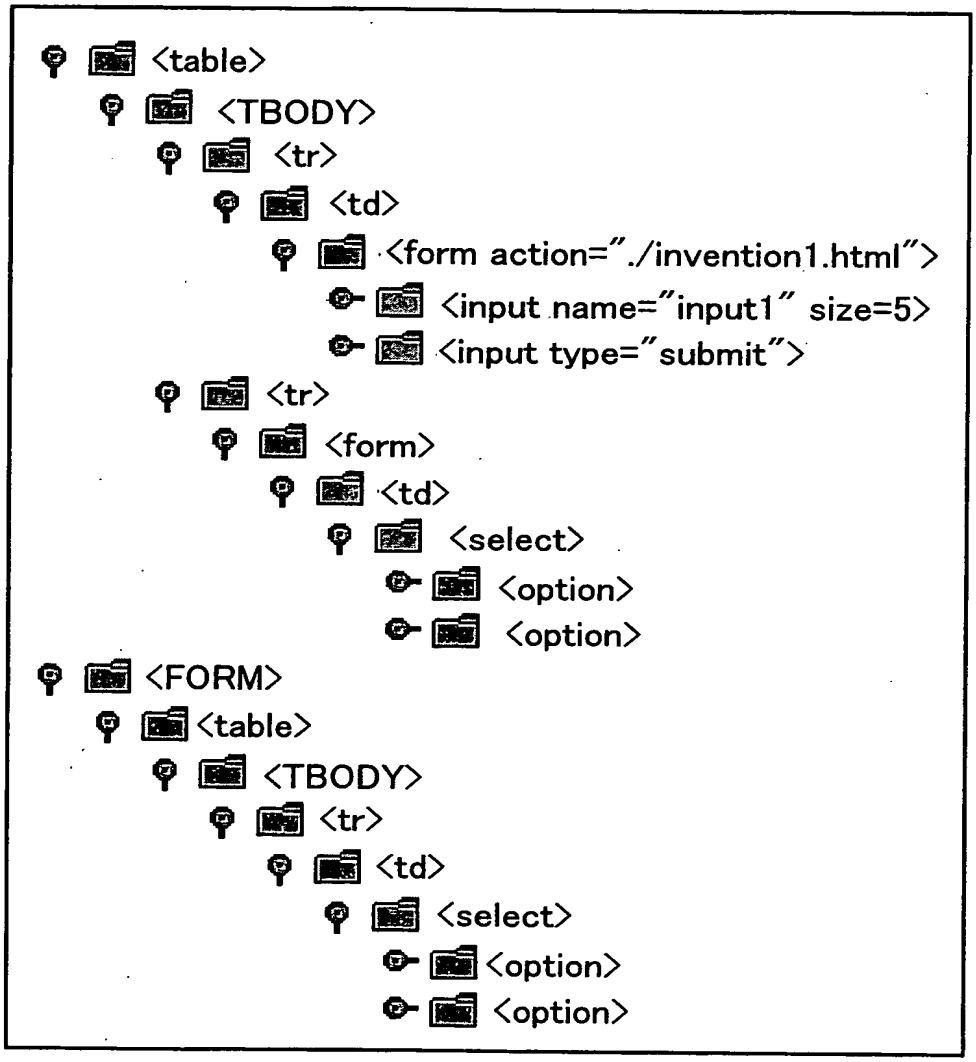
【図８】

FormInserterを適用した場合の文書Aの解析結果



【図９】

FormInserterを適用した場合の文書Bの解析結果

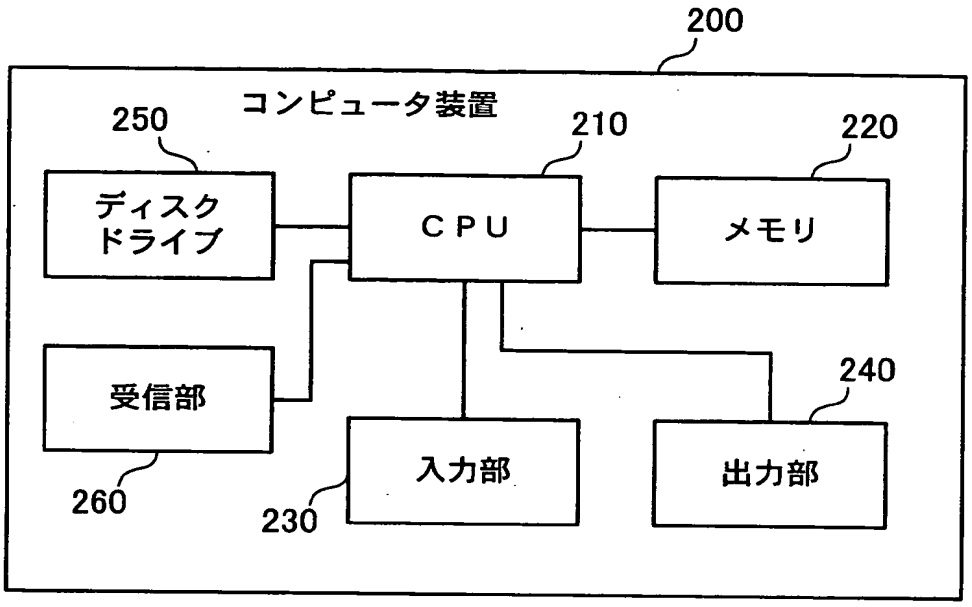


【図１０】

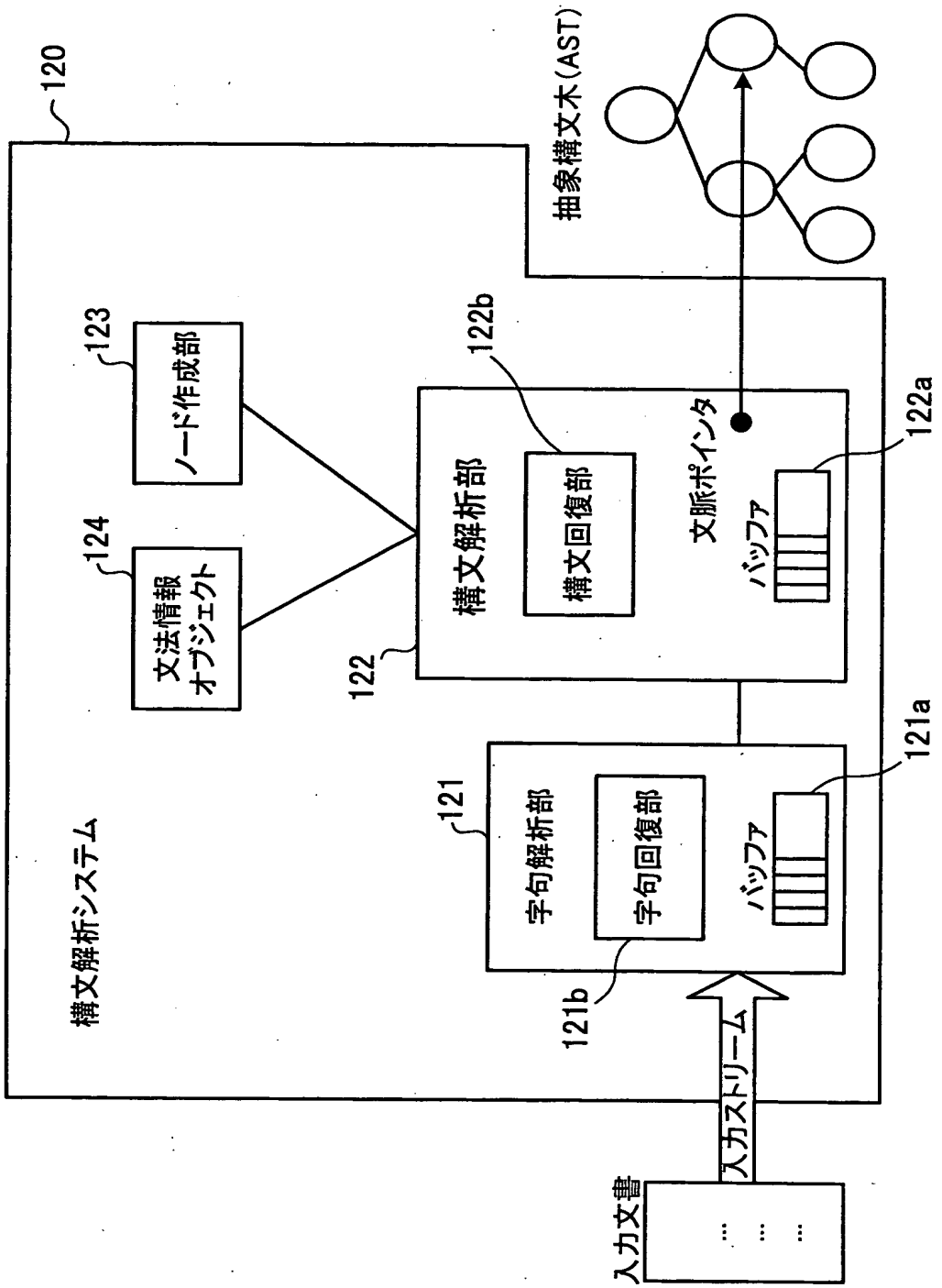
30

文書の識別子	構文回復部
文書Ａ	[FormExpander]
文書Ｂ	[FormInserter]
その他	[]

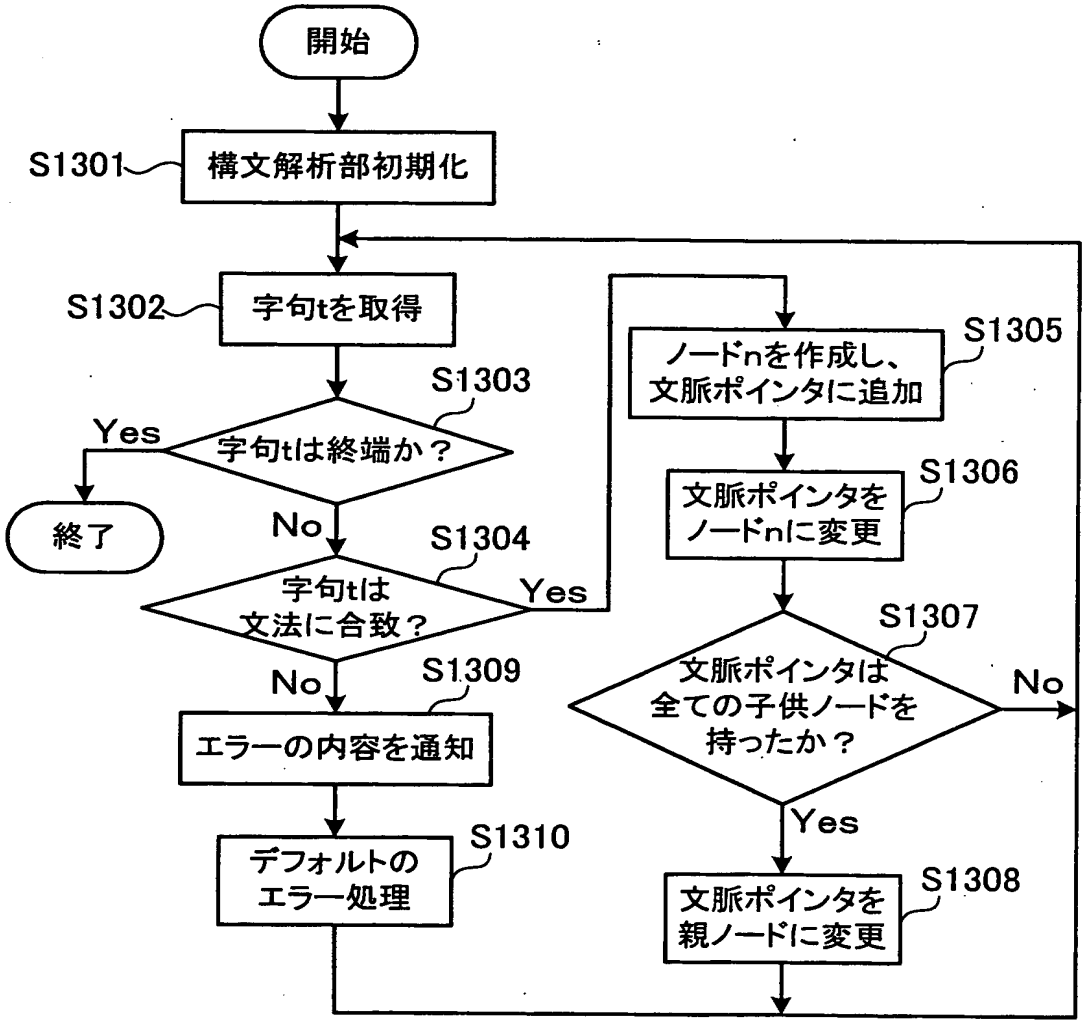
【図１１】


200
コンピュータ装置
250
210
220
ディスク
ドライブ
ＣＰＵ
メモリ
受信部
240
入力部
出力部
260
230

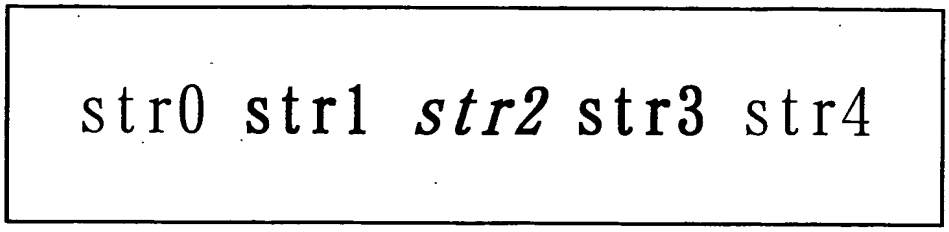
【図１２】


120
構文解析システム
抽象構文木(AST)
123
ノード作成部
124
文法情報
オブジェクト
122b
構文解析部
構文回復部
文脈ポインタ
バッファ
122a
122
121
字句解析部
字句回復部
バッファ
121a
121b
入力ストリーム
入力文書

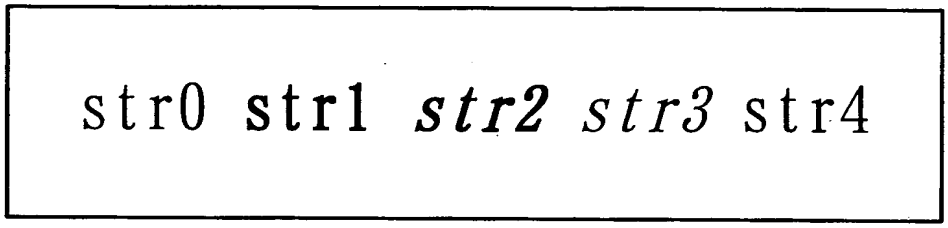
【図１３】


開始
S1301
構文解析部初期化
S1302
字句tを取得
S1303
字句tは終端か？
Yes
終了
No
S1304
字句tは
文法に合致？
Yes
No
S1309
エラーの内容を通知
S1310
デフォルトの
エラー処理
S1305
ノードnを作成し、
文脈ポインタに追加
S1306
文脈ポインタを
ノードnに変更
S1307
文脈ポインタは
全ての子供ノードを
持ったか？
No
Yes
S1308
文脈ポインタを
親ノードに変更

【図１４】



【図１５】



【書類名】　　要約書

【要約】

【課題】　プログラムや文書などのデータ列における字句や構文規則の誤りの種類に応じて複数の回復手段を用意し、選択的に適用して誤りを修正できるシステムを提供する。

【解決手段】　所定の規則に基づいて記述されたデータ列の構造を解析し、規則上の誤りを検出する構文解析部２０と、この構文解析部２０とは独立に設けられ、この構文解析部２０からの依頼に応じて、この構文解析部２０により検出されたこのデータ列における規則上の誤りを修正する回復手段とを備え、この回復手段は、特定の種類の誤りを修正する単純な機能を有する構文回復部２２の集合を含んで構成され、このデータ列における規則上の誤りの種類に応じてこの構文回復部２２を切り換えて使用することにより、このデータ列における種々の誤りを修正する。

【選択図】　　図１

認定・付加情報

特許出願の番号	特願２０００－３２３８２３
受付番号	５００００１３７２２１８
書類名	特許願
担当官	塩崎　博子　　１６０６
作成日	平成１２年１２月　８日

＜認定情報・付加情報＞

【特許出願人】
- 【識別番号】 390009531
- 【住所又は居所】 アメリカ合衆国１０５０４、ニューヨーク州　アーモンク　（番地なし）
- 【氏名又は名称】 インターナショナル・ビジネス・マシーンズ・コーポレーション

【代理人】
- 【識別番号】 100086243
- 【住所又は居所】 神奈川県大和市下鶴間１６２３番地１４　日本アイ・ビー・エム株式会社　大和事業所内
- 【氏名又は名称】 坂口　博

【代理人】
- 【識別番号】 100091568
- 【住所又は居所】 神奈川県大和市下鶴間１６２３番地１４　日本アイ・ビー・エム株式会社　大和事業所内
- 【氏名又は名称】 市位　嘉宏

【代理人】
- 【識別番号】 100106699
- 【住所又は居所】 神奈川県大和市下鶴間１６２３番１４　日本アイ・ビー・エム株式会社大和事業所内
- 【氏名又は名称】 渡部　弘道

【復代理人】 申請人
- 【識別番号】 100104880
- 【住所又は居所】 東京都港区赤坂５－４－１１　山口建設第２ビル６Ｆ　セリオ国際特許事務所
- 【氏名又は名称】 古部　次郎

【選任した復代理人】
- 【識別番号】 100100077

次頁有

認定・付加情報（続き）

【住所又は居所】　東京都港区赤坂５－４－１１　山口建設第２ビル
６Ｆ　セリオ国際特許事務所
【氏名又は名称】　大場　充

次頁無

出　願　人　履　歴　情　報

識別番号　　〔３９０００９５３１〕

１．変更年月日　　２０００年　５月１６日
［変更理由］　名称変更
住　所　　アメリカ合衆国１０５０４、ニューヨーク州　アーモンク　（番地なし）
氏　名　　インターナショナル・ビジネス・マシーンズ・コーポレーション